Modular Neural Network Learning Using Fuzzy Temporal Database

Article · January 2008

DOI: 10.1007/978-1-4020-8735-6_37

2 authors:

Hazra Imran
University of British Columbia
26 PUBLICATIONS **244** CITATIONS
SEE PROFILE

Shadab A. Siddiqui
9 PUBLICATIONS **108** CITATIONS
SEE PROFILE

Modular Neural Network Learning Using Fuzzy Temporal Database

Hazra Imran[1] & Shadab Alam Siddiqui[2]
[1]Department Of Computer Science
Jamia Hamdard , New Delhi -110062
[2]Hughes Systique India Pvt. Ltd.
D-8 Info City II,Sector 33
Gurgaon, Haryana 122022
himran@jamiahamdard.ac.in , shadab@iname.com

Abstract

Despite the availability of several well-known neural network learning algorithms, we have taken the initiative to propose a new mechanism for initial learning and training of a neural network. Our methodology uses fuzzy temporal database as a storehouse of information that would be used to feed the network for learning and perfecting itself.

Keywords

Temporal Database, Modular Neural Network (MNN), Fuzzy Temporal Database (FTD), Fuzzy Temporal Database Learning (FTDL), initial learning.

1. INTRODUCTION

The need for storing imprecise information necessitates the usage of fuzzy database systems. Being able to trace changes in fuzzy sets over periods of time and the ability to access the state of a fuzzy set at a certain point of time requires the concept of temporal database systems. This extension of temporal databases to fuzzy temporal databases requires a database that can store variables belonging to fuzzy sets and their membership values along with a record of all changes in set variables or changes in membership values.

Shadab et al [10] has given Fuzzy temporal database design.In our paper we have used the same design. A simple and general structure for storing temporal fuzzy sets is been used. An example design of the temporal database consists of two tables, which can be shown diagrammatically as:

Member_ID	Set_Name	Variable

(a)

Member_ID	Membership_Degree	Beginning_Time	End_Time

(b)

Figure 1: Fuzzy Temporal Database Design, Example Tables (a) Member_Variable (b) Member_History

For the sake of fulfilling maximum number of constraints of designing a database, every table in a fuzzy temporal database would basically be divided into two sub – tables or normalized, in order to remove redundancy of data. Similarly, the tables shown in figure 1 has also been divided into two such tables: *Member_Variable* and *Member_History*.

Member_Variable table basically identifies the constituent variables of each set, with the provision that each set has multiple elements and each element may belong to multiple sets. Incase the variables are of complex type; they should preferably be stored in a separate table and referenced here through a foreign key. The combination of *Set_Name* and *Variable* should be unique. Each such tuple should have a distinct *Member_ID* - making it a primary key - so that it can be used as a foreign key in another relation. This *Member_ID* identifies the pair (A, x) where A is a set and x, a member variable of A.

Member_History table specifies the dynamic behavior of member variables of a set with respect to time. It must be noted that the

K. Elleithy (ed.), *Innovations and Advanced Techniques in Systems, Computing Sciences and Software Engineering,* 198–202.

history specified here is not specific to either a particular variable or set but to a combination of these. The combination of *Member_ID* and *Beginning_Time* can be considered as the primary key; an alternative approach is to use the combination of *Member_ID* and *End_Time* as the primary key. The column *Member_ID* references the table *Member_Value*.

Membership degree in a fuzzy temporal database is stored as a real number restricted according to the specified range. Storing it as a number enables us to have a much larger range of values and the use of considerably less space vis a vis linguistic variables. Also the generation of linguistic variables is context dependent and can be conveniently performed as demanded by the situation. Also the conversion from a number to a linguistic variable is generally irreversible as the precise value 0.48 cannot be generated from "medium".

The paper is divided into 7 sections. In section 2 we have given an overview of the FTDL (Fuzzy Temporal Database Learning) algorithm. In section 3 , we have the FTDL Architecture. Section 4 deals with the Conceptual FTDL Model. Section 5 and 6 deals with Features and Hitches of FTDL and application of the algorithm respectively. And finally we conclude in section 7.

2. OVERVIEW OF THE FTDL ALGORITHM

Learning in the context of neural network is defined as the process by which the free parameters of neural networks are adapted through a process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameter changes take place.

The basic objectives that a learning algorithm should try to accomplish are:

- Generalization
- Speed of algorithm, and
- Probability of successful convergence.
- Weight should be adjusted on presentation of each sample.
- Learning should be able to capture complex non-linear mapping between input and output pattern pairs, as well as between adjacent pairs in a temporal sequence of patterns.

3. THE FTDL ARCHITECTURE

In this section we introduce the new introductory model that is used for the proposed FTDL algorithm. Before that, the described example of a fuzzy temporal database in the figure 1 is integrated into one for ease of use and its diagrammatic representation. Various explored factors for the design is also discussed in this section. Integration of FTD and modular neural network architecture to be used is also discussed.

Event	Membership_Degree	Beginning_Time	End_Time

Figure 2: Compact FTD Design for FTD Learning

The above figure shows the compact design of a fuzzy temporal database. This has been designed to ease the explanation of the use of FTD in the modular network architecture. In fact, the *Event* field of the above-described architecture is an embedded database table. When this table is disintegrated, it forms two tables as in figure 1, the primary key of the first table is an attribute of the second table and acts as a foreign key. The combination of this foreign key along with the *Beginning_Time* attribute is the primary key of the second disintegrated table.

The value in the *Event* field specifies the changes within the environment where the database is used. The *Beginning_Time* and the *End_Time* attributes together define the time period for which the event was stable at a particular value i.e. it did not experienced any signs of fluctuations or any value changes in the embedded database.

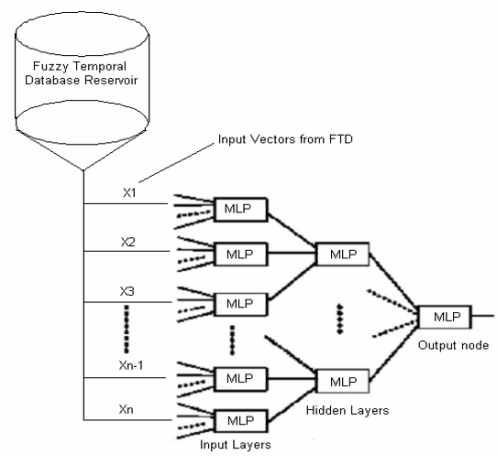

Figure 3: Preliminary Architecture for FTD Learning

The above diagram is the basic structure resulting from the integrated fuzzy temporal database (FTD) and the modular neural network (MNN). This diagrammatic representation is designed for the fuzzy temporal database learning (FTDL) algorithm.

As each module in the modular neural network is entitled to perform their respective task, thus the input to every module would be a different FTD, related to the task to be performed by the module of the network. This is one of the drawbacks to the proposed approach i.e. for every module a different FTD should exist or in other worlds the number of modules of a particular MNN is restricted by the available FTDs of the system in question.

In the figure 3, every MLP can be considered as a module of the entire system but at a larger scale the entire system depicted in the same figure can be considered as a module. This is totally based on the size and complexity of the system in question and the environment within which it is to be implemented.

The input vectors (X_1……X_n), in the above diagram is a collection of n FTDs, which acts as input to the modules of the modular neural network. This states that every n is a FTD and it is used as an input vector to the input module of the MNN and result of the processing of these input modules is fed to the subsequent modules in the hidden layers, which perform their intended task. The number of modules in the next layer is always less than the previous layer because a set of output from previous layer is fed to a single module in the next layer. This phenomenon of module integration carries on till the final output from the output module.

4. CONCEPTUAL FTDL MODEL

As stated earlier, this learning algorithm is still in its initial stages of development and the concept too is at a very immature stage. In this section we have described and discussed the theoretical aspect of the proposed model

Briefly, the FTDL algorithm falls under the broad category of supervised learning algorithms in neural networks. The mechanism is thought of under the influence of supervised form of learning and thus would imitate the basic characteristics of this mode of learning.

Very simplest form of fuzzy temporal database learning (FTDL) algorithm can be represented in the mathematically as:

$$\Delta w(n) = f(\eta)\left[\left(X_{mn}\right)\zeta\left(Y_{(m-1)(n-1)}\right)\zeta\left(\mu_m(x)\right)\right]$$

Figure 4: Mathematical formula for FTD Learning

FTDL learning, thus is a function of combination of current input parameters, output from the previous stage and fuzzy membership value of the current tuple from the input FTD with respect to the rate of learning of the modular neural network, within the given environment.

The variables of the FTDL formula prescribed above and their meaning with respect to the proposed algorithm are:

$\Delta w(n)$ = change in weight of node n.

η = rate of learning of the FTDL algorithm.

X_{mn} = m^{th} input tuple to the node n form the FTD n corresponding to the n^{th} modular node.

$Y_{(m-1)(n-1)}$ = Output of the input $X_{(m-1)(n-1)}$

$\mu_m(x)$ = fuzzy membership value of the tuple m from the input FTD.

ζ = any operation between any two variables required by the environment or the system.

This is the very basic and generic mathematical structure of the fuzzy temporal database learning algorithm. The relation and the dependencies amongst the variables are still to be decided.

Initialization of a neural network is a very important aspect of a NN system. In FTDL approach of learning, initialization of the modular neural network can be easily done by using the FTD tuple of the FTD to be used as input to the module in question, with the minimum membership degree value (only a possible approach). Weight dynamics and weight space problem is also sorted and eased. We can assign initial weights to the system with the help of a function with the membership value as its required parameter. Once initialized, the MNN keeps on learning using the defined formula and the designed rules.

One of the possible relationship between the change in weight ($\Delta w(n)$) and the membership value ($\mu_m(x)$) in the given formula, is that they would be inversely proportional to each other. This is because the greater the membership value of the tuple, the lesser the difference would be between the actual output and the desired output from the modular neural network.

Another aspect explored with this design is that, the order in which we give in the input to the network should be in the ascending order with respect to the membership value of the input FTD. This is because, starting from the tuple having the lowest membership value and going to the maximum ones would result in step by step learning and finally reaching near or to the desired output. Indirectly this states that, the FTD should be sorted before it is fed to the module of the MNN. This sorting should be done such that the passed by events are arranged in ascending order and the event in progress (i.e. the event having End_Time value as "now") should not be considered as input tuple. This criteria also holds for the passed events, which did not had a membership value due to any unspecified reasons.

5. FEATURES AND HITCHES OF FTDL ALGORITHM

Generally a neural network is initialized using random number values. One of the most important features of this network is that no random number guessing is required for initializing and activating the modular network. We can initialize the network by looking at the membership value of the tuples. With this methodology, weight initialization becomes an easier task. For this particular aspect, all membership values are considered and a weight space is formulated, from the formulated weight space, using a function defined according to the environment, the initial weights can be taken.

When any MNN is fully trained with the FTDL algorithm, then with very slight modifications, we can use this very MNN to get a fuzzy temporal database by inputting only temporal databases. This is an important feature of this algorithm and can be viewed as an application of the FTDL algorithm.

One of the major drawbacks of the FTDL algorithm is that its applicability is limited by the availability of the fuzzy temporal databases. In other words, any modular neural network wishing to use FTDL should posses a bank of fuzzy temporal databases related to the system for which the network is intended to be. When any MNN to be designed, wants to use FTDL, it should first look for the FTDs and the number of modules in the entire MNN would also be restricted by the number of FTDs available for the system.

6. APPLICATIONS OF FTDL ALGORITHM

Although this work is still in its initial phases, but certain basic applications of this algorithms are thought of and are described in this section.

One of the most important areas where this algorithm would be of great use is in the predicting the output of an event. Looking at the features of this algorithm, when the network is fully trained then it can be useful in getting the membership values of a tuple whose $\mu_m(x)$ is not known for the FTD, which was used earlier as an input to a module of the MNN. Extending this feature, we can utilize the trained MNN for converting simple temporal databases to fuzzy temporal bases.

Any applications involving fuzzy temporal databases, when they are incorporated with neural networks would find this learning technique useful and easier as compared to other neural network learning algorithms. Several other applications of this mode of learning to the modular neural networks are possible and would exist but due to time limitations they cannot be explored and discussed in this work.

7. CONCLUSION

The proposed method for initial learning of the modular neural network is only theoretical. No experimental verifications have been made because the algorithm is still to be implemented and simulated. Future work in this domain will be focussed on its possible applications, areas of its applicability are also been considered as future endeavors. Searching for the domain of problems is a task to be accomplished.

References

[1] Bart Kosko, "Neural Networks And Fuzzy Systems", Prentice Hall, 1992.

[2] Bart L.M. Happel and Jacob M.J. Murre, Design and Evolution of Modular Neural Architectures. Neural Networks, Vol. 7, No. 6/7, Pages 985 - 1004, 1994.

[3] Becker, S., Unsupervised Learning procedures for Neural Networks, International Journal of Neural Systems, Vol. 2, pp 17 – 33, 1991.

[4] C. J. Date, An Introduction to Database Systems, 7th edition, Pearson Education Asia Pte. Ltd, 2000.

[5] George J. Klir and Bo Yuan, Fuzzy sets and Fuzzy Logic: Theory and Application, Prentice Hall Inc., 1995.

[6] James A. Anderson and Edward Rosenfeld, Talking : An Oral History of Neural Networks, MIT Press, 2000.

[7] Laurene Fausett. Fundamentals of Neural Networks. Architectures, Algorithms, and Applications. Prentice-Hall. New Jersey, 1994.

[8] Mohamad H. Hassoun. Fundamentals of Artificial Neural Networks. Prentice Hall of India, 2002.

[9] Ramez Elmasri and Shamkant B. Navathe, Fundamentals of Database Systems, 3rd edition, Pearson Education Asia Pvt. Ltd., 2000.

[10] Shadab A. Siddiqui, Javed A. Alvi and A.Q. Ansari, A Generalised Fuzzy Temporal Database System With SQL Compatibility, 8th WSEAS Conference on Computers, pp. 222 – 226, 2004.

[11] Shadab A. Siddiqui, A.Q. Ansari and Shikha Agarwal, A Journey through Fuzzy Philosophy, PRANJANA, a Journal of Management Awareness, Vol. 6, No. 2, pp. 29 – 33, 2003.

[12] Simon Haykin. Neural Networks - A Comprehensive Foundation. New York, 1994.

[13] W. Au and C. Chan, Mining changes in association rules: a fuzzy approach, *Fuzzy Sets and Systems* **149**, pp. 87–104,2005

[14] Zadeh, L.A., Fuzzy Sets, Journal of Information and Control, 8, pp. 338 - 353, 1965.

[15] Zadeh, L.A., Fuzzy Sets, Information and Control, 8(3), pp. 338 - 353, 1965.